C

Chapter 3 Architecture Frameworks for Client/Server and Netcentric Computing

THE NEED FOR ARCHITECTURES: INSURANCE AGAINST RISK

At the heart of systems development, the use of architectures provides *insurance*: insurance against the complexities of development and maintenance, against the obsolescence of technologies, against the possibility that all the parts of a solution may not work together. Architectures are the master plans that ensure that the solution will work.

This notion implies that risk is involved, and that is so. In client/server and netcentric environments, a number of risks are generally present.

More Complex Development and Maintenance

A number of factors contribute to the complexity of client/server and netcentric solutions:

- Client/server applications incorporate sophisticated graphical user interfaces (GUIs). GUIs are usually event driven rather than hierarchical. They are interactive and require more complex logic than traditional terminal (e.g., 3270) style interfaces.
- Client/server and netcentric applications have to "cooperate" with other applications. Communication code must establish communication connections, ensure that messages are sent and received correctly, manage errors, and handle any required data translation. Care must be taken that programmers and designers have these skill sets.
- The skills required for development, installation, and support of netcentric systems may be difficult to find.

More Difficult Operations Support

Operations support for netcentric solutions is more difficult than for traditional systems. The increased complexity of operations support, including hardware and software configuration management, is directly related to the number and location of distributed nodes. If a system has 100 remote nodes, it is more difficult to ensure that they are at the same software and hardware versions than it is with two local nodes.

In addition, data backup/restore must now occur at multiple locations, and support for hardware, software, and communications problems must also be provided locally at multiple sites.

More Complex Data Security

When data are distributed, protecting that data becomes more difficult. Intelligent workstations are inherently less secure than minicomputers and mainframes. The effort required to maintain an equivalent level of data security, therefore, increases.

New Distributed Data Update and Refresh Strategies

Most client/server systems incorporate multiple copies of the same data. This requires logic to ensure that data values in each of those copies are consistent. For example, if a user working off server A wants to change a "balance due" field, how and when will this change be reflected on servers B and C?

Increased Susceptibility to Viruses and Malicious Users

Again, this risk is directly proportional to the number of nodes in a distributed system. Each workstation is a potential point of entry for a virus or a malicious hacker.

Higher Communications Loads

Netcentric applications must communicate with each other and with other applications, typically legacy systems. This is accomplished over communications networks. For a networked system to work well, accurate estimates of the amount of network traffic must be determined. This is often difficult because, as the knowledge and popularity of newly released applications increase, application use (and network traffic) increases. Applications designed with communication speeds in mind may, therefore, end up being "communications bound." In addition, there are not many tools available that model new-age computing communication loads.

Missed Opportunities

Because netcentric systems are comprised of hardware and software that are continually being improved, it is often difficult to stop waiting for enhancements. Many development teams become paralyzed, waiting for the next release of some component that promises to facilitate the installation process or enhance the final product.

Lack of a Standard Operating Environment

There are many popular operating system and window manager options that can be used to develop workstation applications. The risk is in choosing a combination that ends up with little or no support in the long run and requires future migrations of applications and data.

Increased Complexity of User ID and Password Strategies

Because netcentric solutions require the use of multiple computers, user ID and password strategies become more complex. For example, a security system on one computer may require password changes more frequently than another, or maximum and minimum password lengths may conflict on different systems. Even if these issues are not present, the maintenance of security information on multiple platforms is difficult.

THE BENEFITS OF ARCHITECTURES

The risks just discussed illustrate the need for architectures as crucial aspects of client/server and netcentric systems development. What is an architecture?

An architecture is a proven mechanism and an approach that can be used to isolate and mitigate the risks of delivering applications now and into the future.

According to the Gartner Group, an architecture is "a formal specification of how a computer solution will be organized." Gartner sets forth seven characteristics of a successful architecture:

1. Delimitation of the problem to be addressed.
2. Decomposition of the solution to components with clearly assigned responsibilities.
3. Definition of interfaces, formats, and protocols to be used between the components. These should be sufficiently clear and robust to permit asynchronous development and ongoing reimplementation of the components.
4. Adequate documentation to permit compliance by implementers.
5. An auditing mechanism that exercises the specified interfaces to verify that specified inputs to components yield specified results.
6. An extendibility mechanism to enable response to changing requirements and technologies.
7. Policies, practices, and organizational structures that facilitate adoption of the architecture.

In the netcentric environment, an architecture is used to define how a system is structured and how the various components of the system interact. In a netcentric computing environment, there are more components and many more interactions that make an architecture even more important.

Organizations that have carefully implemented, delivered, and utilized these architectures have realized some of the following benefits:

1. *Better productivity, and less "reinvention of the wheel."* Architectures can abstract common requirements and approaches from applications and can eliminate having to identify, analyze, and implement them for each application. This improves developer productivity and the quality of the final product.
2. *Consistent, reliable, high-quality applications.* The framework provided by an architecture encourages applications to be built in a consistent fashion or structure, to deliver consistent behavior, and to work with a consistent interface (both to users and other applications), resulting in a system easier to build, use, and maintain.
3. *Rapid delivery of business solutions.* By providing a consistent external interface, an architecture simplifies integration of applications and facilitates rapid delivery of new solutions. This is achieved through the use of standard architecture components, adherence to standards, and the availability of the necessary tools, techniques, and training.
4. *Reduced impact of changes to underlying products and tools.* Because an architecture incorporates "layers of isolation," new products and tools can be more easily integrated into a system. Changes in one element of the architecture are less likely to affect other architecture elements.
5. *Better integration of business applications within and between organization business units.* By providing consistency of architecture components within and across an organization, the opportunity to build applications that have a higher degree of integration is greater. This should facilitate the exchange of critical information across the company.
6. *Isolation of users and applications developers from the complexities of the underlying technologies.* By having a standard architecture that includes a standard set of tools with a consistent interface, users and developers are not required to know the details of the platform technologies (i.e., the operating system, database, and network). Additional technology components could be added in the future with minimal additional training for the users.
7. *A consistent, standard development framework.* An architecture provides a framework for analyzing the requirements of a system or application. It can help business applications developers by providing

a structure from which to work. In a netcentric environment, the requirements of a GUI, distributed data, and distributed processing contribute to the complexity of the solution. Moreover, these requirements have many interdependencies. Without an architecture to help structure the problem, it is easy for applications developers to become overwhelmed by technical issues and spend insufficient time on the business problems they are there to solve.

8. *A common background for IS personnel.* In addition to providing a common approach for building systems, an architecture provides a common means of describing systems and a common language. As a result, IS personnel are more easily interchanged and cross-trained, providing more flexibility in the management of the organization.

This chapter will move from a high-level description of an overall architecture — what is called an Enterprise Information Architecture — to a summary of the primary technical architectures discussed in this book: the execution, development, and operations architectures for client/server and netcentric computing solutions. More detail on each of these architectures — their services and subservices — is provided in subsequent chapters of Section II.

THE ENTERPRISE INFORMATION ARCHITECTURE (EIA)

What are the components of an effective architecture? The Enterprise Information Architecture (EIA) framework provides a starting point for understanding what is meant by the various architectures under consideration. The EIA framework contains seven layers (Exhibit 1).

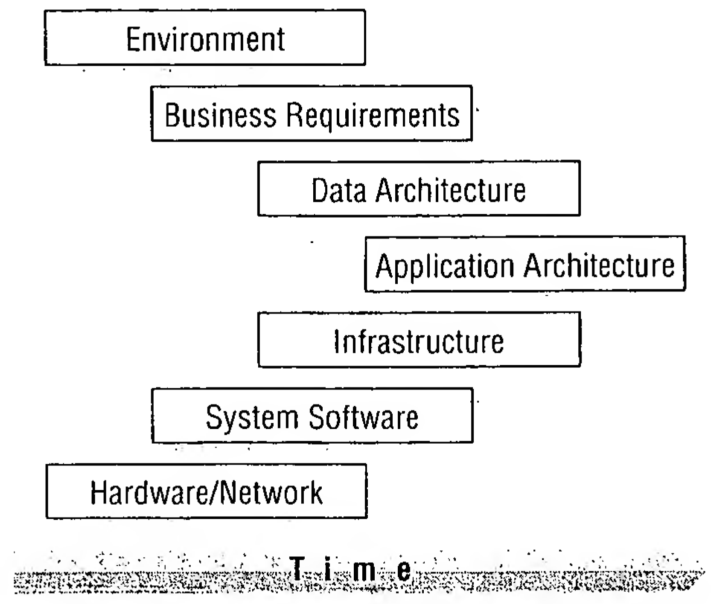


Exhibit 1. Enterprise Information Architecture (EIA).

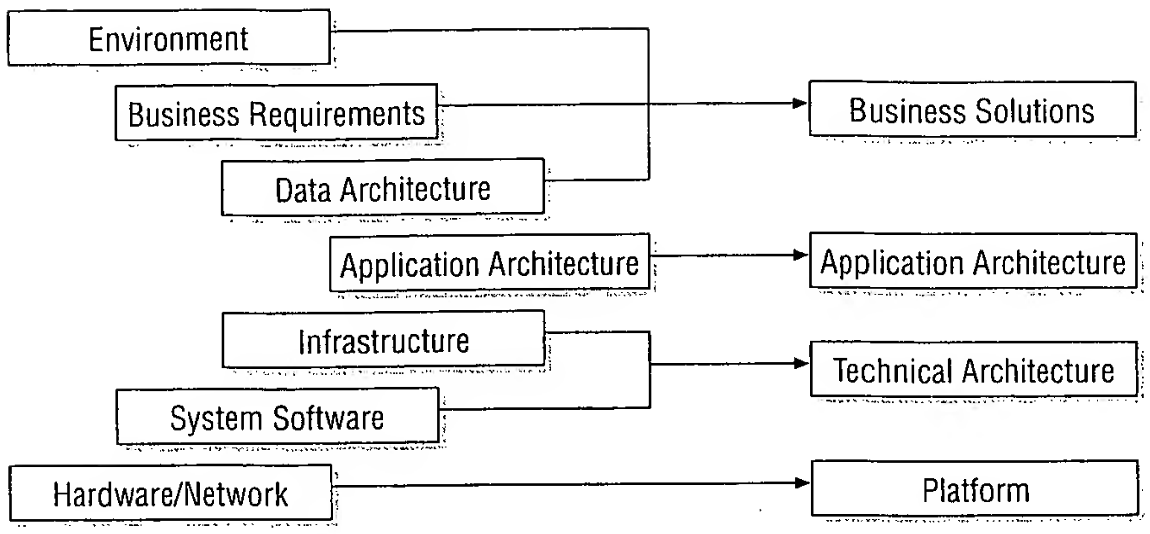
- The *environment* layer includes those factors that influence the business requirements and technical layers. These factors may be either internal (e.g., profitability) or external (e.g., government regulation and market competition).
- The *business requirements* layer addresses the business needs of the organization. Both the environment layer and the business requirements layer are mainly concerned with business-level processes, strategies, and directions. The layers below are mainly concerned with the information technology to support the business. The business requirements give key input and guidelines on how to define the lower layers. The link from business requirements to the information technology layers is crucial to a successful EIA.
- The *data architecture* layer consists of a high-level data design that describes the structure of an enterprise's data needs in terms of entities and relationships between entities. The structure and relationships of data entities can be used to define the basic relationships of business functions and applications.
- The *applications architecture* layer defines the applications that must exist to support the business functions and their relationships. It also addresses any issues about distributed data processing.
- The *infrastructure* layer deals with those components of an architecture that can be used by multiple applications and that are developed and maintained within the enterprise. Usually, these common technical components help support the applications architecture. This layer also includes the infrastructure of the organization that manages the architecture definition and design and its technical components.
- The *systems software* layer encompasses the software and standards obtained from and maintained by outside vendors (e.g., a database management system.)
- The *hardware/network* layer deals with central processing units, local area network (LAN), wide area networks, and other hardware and physical network components of the environment.

Redefining the Enterprise Information Architecture

For purposes of this volume, these components can be grouped into four categories of architecture (Exhibit 2).

Business Solutions Architecture

Because this chapter does not focus on business specifics, the top three levels can be grouped into a business solutions architecture. It is important to remember, however, that, when it comes time to decide what technical architecture to use, many of the answers are found by looking at the business solutions architecture. The decisions made for the application and data architectures drive the requirements of the technical architecture and



Exhbit 2. EIA Model Redefined.

platform. At the same time, the constraints of the technical architecture and platform can also shape the application architecture and the business solutions that are possible.

Applications Architecture

The applications architecture layer can be defined here as those services that perform business functions on the computer. It represents the components that provide the automation support for a business function or activity in the business process (but does not include the platform and cross-application architecture). For example, a manufacturer's sales and marketing system application architecture could include sales tracking applications and the distributed data architecture to support both networked sales offices and mobile sales people.

Technical Architecture

The Infrastructure and System Software layers are combined to form the technical architecture. The technical architecture is where the buy decisions of the system software marketplace are combined with the build decisions for the needs of specific applications. We treat these as one architecture by incorporating these two concepts. The technical architecture is comprised of the execution, development, and operations architectures, which are discussed subsequently.

Platform Architecture

The final layer in the EIA model is the platform architecture layer. It is often described as "the things you can see." The netcentric platform architecture provides a framework for selecting the platform components required: the servers, workstations, operating systems, and networks. This framework represents the overall technology platform for the implementation and deployment of the execution architecture, development architecture, operations architecture, and, of course, the applications.

THE TECHNICAL ARCHITECTURE

Because of its relative importance in client/server and netcentric implementations, the technical architecture will be discussed in some detail in the remainder of this chapter. The technical architecture consists of the infrastructure and systems software layers, as discussed previously. The differentiation between them is primarily a question of make vs. buy, that is, a key decision for organizations intent on "building an architecture" is how much they want to build vs. how much they can simply buy from preexisting sources. An organization can choose to build a great deal, thereby making the architecture very close to what it wants. That means that there is a great deal of logic being built by the shop.

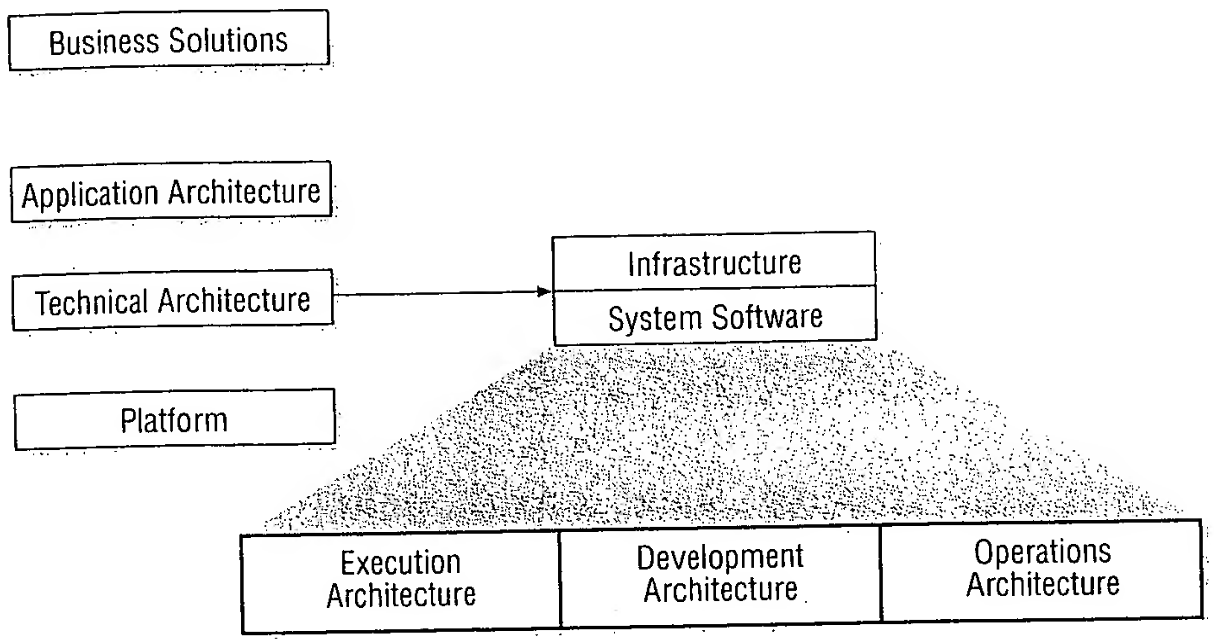
Alternatively, the organization can choose to buy most of what it wants. To the extent that business or application demands make it necessary for the tools to be integrated, developers can then do simple assembly, or gluing together, of the pieces. The decision for most organizations depends on balancing demands. On the one hand, the organization has a large front-end commitment to build and an ongoing commitment to maintain an infrastructure architecture; on the other hand, the organization has a tool that is exactly what it wants.

Over the years there has been a tendency to buy rather than make. This is especially the case as the market matures with more technical entrants. It is practical for IS organizations to build technical architecture components only when essential. By purchasing rather than building, they can then more easily apply their strong skills in the applications architecture business.

Components of the Technical Architecture

The technical architecture layer can in turn be broken into three primary components, execution, development, and operations (Exhibit 3).

- An *execution* architecture describes the components required when an application executes.
- A *development* architecture describes the components required to create the execution architecture.
- An *operations* architecture describes the components required to operate and manage the system.



Exhibt 3. Three Components of a Technical Architecture.

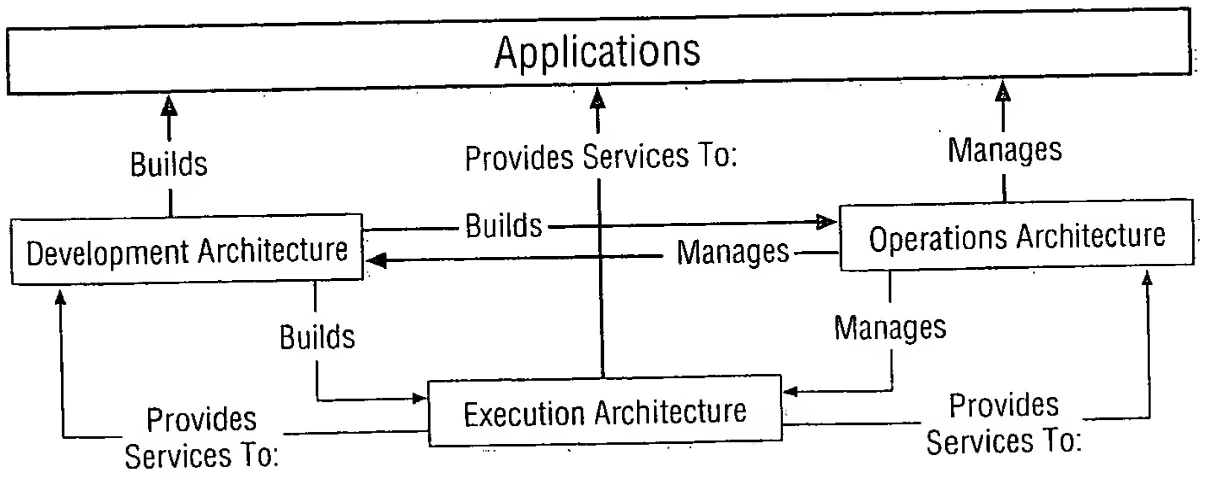


Exhibit 4. Relationships Among the Technical Architectures.

These architectures must be flexible enough to accommodate a wide range of technologies, but they must also be structured enough to provide valuable guidelines and to ensure that interoperability is available where it is required. Exhibit 4 illustrates the relationships among the execution, development, and operations architectures.

The remainder of this chapter will provide an overview of these technical architectures. Because of its relative importance in the design and delivery of netcentric solutions, the execution architecture will be discussed last, and in a great deal more detail.

DEVELOPMENT ARCHITECTURE

The development environment is the production environment for one or several systems development projects as well as for the maintenance efforts. Thus, it requires the same attention as a similarly sized end-user execution environment.

The purpose of the development architecture is to support the tasks involved in the analysis, design, construction, and maintenance of business systems as well as the associated management processes. It is important to note that the environment should adequately support *all* the development tasks, not just the code/compile/test/debug cycle. Given this, a comprehensive framework for understanding the requirements of the development environment should be used.

Another reason for the comprehensive framework is that it is important to get the development environment right the first time. Changing the development environment when construction is fully staffed may entail serious disruptions and expensive loss of productivity.

Experience has shown that, within the same medium- to large-size project, with the same people, moving from a poor to a good development environment, productivity can be improved by a factor of ten for many tasks. The improvements come in two categories:

- The elimination of redundant and non-value-added tasks
- The streamlining of useful tasks

While it seems intuitive that most tasks can be streamlined, the following list gives a few examples of redundant tasks that must be eliminated:

- Analysis to determine how to merge the uncoordinated changes applied by two programmers to the same module.
- Reentry of the source code for and retesting of a module, which was accidentally deleted.
- Recurring discussions about "what a design packet should contain" or "what constitutes good programming style in a particular context."
- Repeated design, coding, testing, and maintenance of very similar logic (e.g., error handling, date conversion and manipulation, main structure of a module).
- Searching for the manuals of a particular productivity tool to find information.
- Remigration to system test of a cycle because the impact analysis for a change request was incomplete.
- Requesting support from another team (e.g., environment support, information management) and waiting unnecessarily for a response.

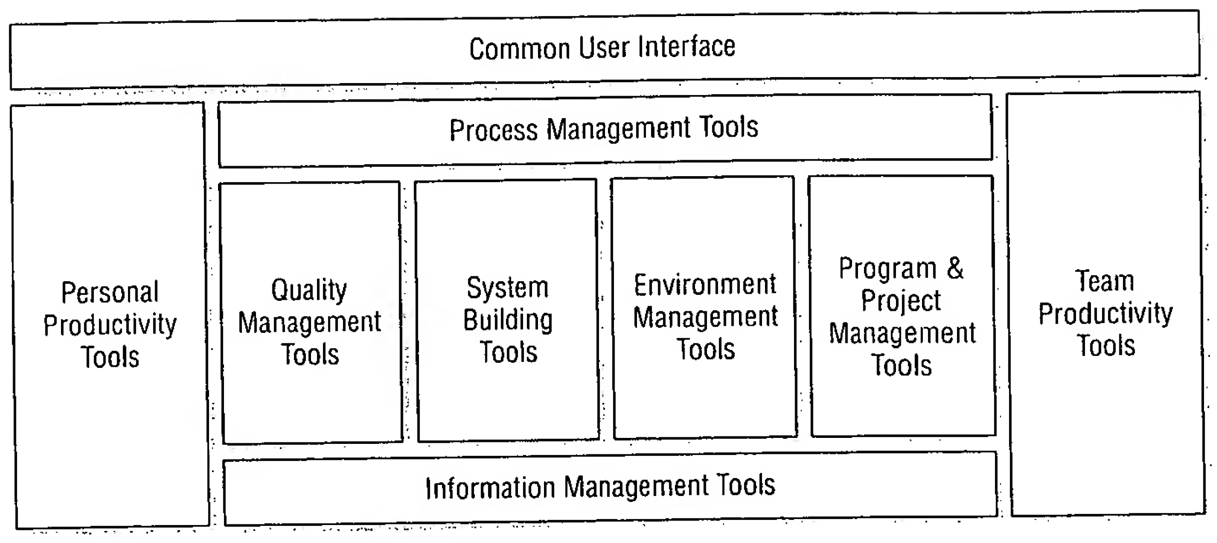

Common User Interface
Process Management Tools
Personal Productivity Tools
Quality Management Tools
System Building Tools
Environment Management Tools
Program & Project Management Tools
Team Productivity Tools
Information Management Tools

Exhibit 5. Development Architecture.

On a smaller project, these problems can be solved using a brute force approach. This becomes very expensive as the project grows, and finally, impossible. A well-designed development environment becomes important as the project team reaches 20 to 30 people and is absolutely critical with a project size of more than 50 people.

The investment needed to design, set up, and tune a comprehensive, good development and maintenance environment is typically several hundred man days. Numbers between 400 and 800 days are commonly seen, depending on the platforms, target environment complexity, amount of reuse, and size of the system being developed/maintained. This investment warrants the following considerations:

- *This effort is large enough to justify work that will make it more efficient.* Among the factors that affect the effort, reuse is the most apparent. These guidelines, together with the parallel project to instantiate the model, constitute a step toward greater reuse.
- *The effort is large enough to require a cost/benefit analysis.*

Exhibit 5 is the model used throughout this book to describe the development architecture. The components of the development architecture include the following.

Common User Interface Tools

Common user interface tools provide a common launching place for all the tools in the development environment to make it appear more integrated and consistent. This is the simplest level of integration, in that all the tools are presented to the developer via a single view of the entire environment. Tools that support the common user interface are known as window managers (e.g., Microsoft Windows, Presentation Manager, and Motif).

Process Management Tools

Process management tools integrate the development environment by providing tool-to-tool communication and workflow management. Tool-to-tool communication integrates tools by enabling information in the form of short messages to be passed from one tool to another. Workflow management integration builds the development methodology and process into the tool environment. Workflow management enforces the correct sequencing of tasks and tools. Process integration is often implemented through the use of integration frameworks or through custom coding of interfaces.

Personal Productivity Tools

Personal productivity tools are a collection of software applications that enhance the development environment for the individual developer. These applications are typically integrated suites of PC software that allow the developer to work on the workstation independent of the development server or mainframe to complete tasks such as analysis and documentation. These tools are basic office automation software and include spreadsheet software, word processing software, graphics software (e.g., drawing, diagramming, and presentation), and personal calendar software.

Quality Management Tools

Quality management is a management discipline that promotes a customer satisfaction focus and continuous improvement. Quality management tools support the planning and measurement of quality. These tools include quality function deployment tools, measurement and metrics tools, statistical process control tools, and continuous improvement tools.

System Building Tools

System Building tools comprise the core of the development architecture and are used to design, build, and test the system. All the system building tools must be integrated and share development objects appropriately. These include

- Analysis and Design tools
- Reverse Engineering tools
- Construction tools
- Testing tools
- Configuration management tools

Environment Management Tools

A netcentric development environment is complex and sophisticated. It supports many different functional and technical requirements (illustrated by the Execution Architecture), many different development teams, and tools from many different product vendors and often must support projects in different stages of the development life cycle. These tools monitor performance, provide help desk support, manage and distribute changes to the development environment, administer the environment, and track and plan development environment capacity.

Environment Management tools include

- Service Management tools
- Systems Management tools
- Managing Change tools
- Service Planning tools

Program and Project Management Tools

Program and project management are usually differentiated by the size of the effort; programs are typically composed of more than one project. Similarly, the program and project management tools are differentiated by the ability to support multiple projects, complex functions, and adequate performance when supporting multiple concurrent projects.

Program and project management tools provide many key features that assist project planners in planning, scheduling, tracking, and reporting on project segments, tasks, and milestones.

These tools include

- Planning tools
- Scheduling tools
- Tracking tools
- Reporting tools

Team Productivity Tools

Team productivity tools are used to make the work cell and project team as a whole more productive. Instead of the software residing on the individual's PC or workstation, these tools typically are LAN based and shared by the project members. These tools are focused on enhancing communication and information sharing.

These tools include

- E-mails
- Teamware
- Publishing tools
- Group calendars
- Methodology browsing tools

Information Management

Information management of the development architecture is provided through an integrated development repository. At this level of integration, tools share a common repository of development objects, design documents, source code, and test plans and data. Ideally, the repository would be a single database with an all-encompassing information model. Practically, the repository must be built by integrating the repositories of the different development tools through interfaces. Tool vendors may also build part of the integrated repository by integrating specific products.

The repository includes

- Folder management
- Repository management

OPERATIONS ARCHITECTURE

An operations architecture is a combination of tools, support services, procedures, and controls required to keep a production system up and running well. It differs from an execution architecture in that its primary users are systems administrators and production support personnel. Exhibit 6 shows the framework used throughout this book to illustrate the operations architecture. It depicts a set of tools supporting the execution and development architectures.

The major tool categories of the operations architecture include the following.

Software Distribution

Software distribution is the automated delivery to, and installation of, applications and systems software on servers and end user devices (e.g., workstations, kiosks, etc.). This can be for an organization's internal computing environment as well as for its extended one, i.e., its business partners and customers. The architectural support required to support software distribution is largely driven by the numbers of workstations, servers, and geographic locations to be served.

Configuration and Asset Management

To manage a netcentric environment successfully, one must have a solid understanding of *what* is *where*, and one must maintain rigor in the change control procedures that govern modifications to the environment. Configuration and asset management information that may need to be tracked includes such details as product licensing information, warranty information, vendor names, logical and physical device information (such as total capacity and current utilization), product configuration tracking, software

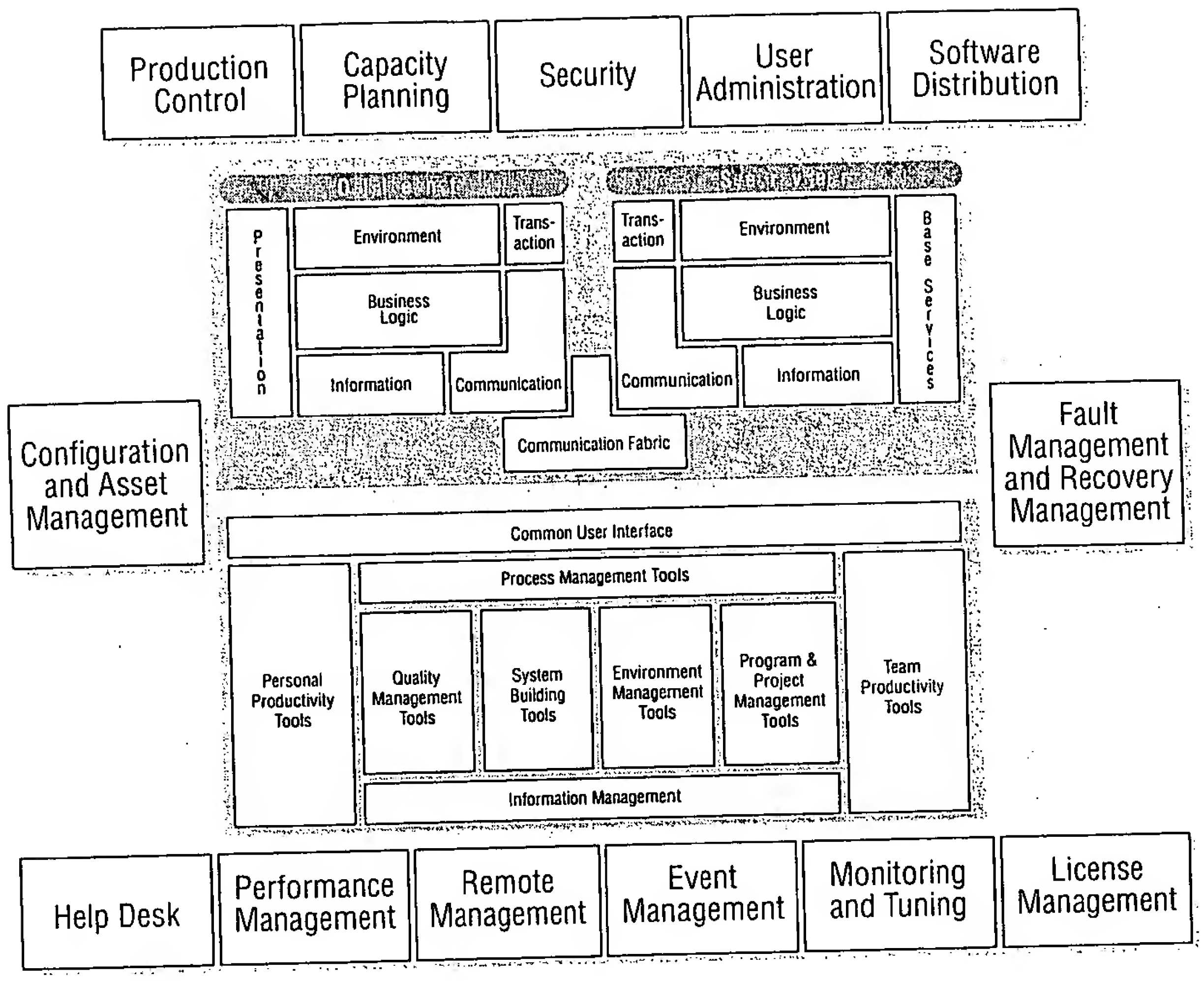


Exhibit 6. Operations Architecture Framework.

and data version levels, network configuration parameters, physical location, and perhaps accounting information.

Fault Management and Recovery Management

The fault management services of an operations architecture assist in the diagnosis and correction of system faults. Faults may include network-, server-, workstation-, or even application-level faults. Fault diagnosis may require services for isolation; viewing of host, server, and workstation error logs; and determining the software and data versions and configurations of affected machines.

Capacity Planning

Capacity planning tools focus on components of an environment such as the network, physical space, and processing power to understand the need to change the capacity of those components based on organizational changes. The tools typically focus on components that are considered to be heavily sensitive to changes in computing resource usage. The tools may use historical management data combined with estimates for growth or changes to configuration to simulate the ability of different system configurations to meet capacity needs.

Performance Management

Performance management is more difficult because of the lack of tools to assist with performance in heterogeneous environments. Performance is no longer confined to the network or to the central processing unit. Performance needs to be viewed in an end-to-end manner, accounting for all the factors that affect the system's performance relative to a user request.

License Management

In addition to guaranteeing compliance with software licensing agreements, license management provides valuable information about which people and how many people are actually using a given software product.

Remote Management

Remote Management tools allow support personnel to "control" a user's desktop over a network so that they do not need to be physically present at a workstation to diagnose problems. Once control of the desktop is established, screen updates for the controlled desktop are displayed at both locations. The support person is then effectively sitting at the workstation he/she controls and can do necessary diagnostics.

Event Management

In addition to hardware devices, applications and systems software also generates events. Common event-handling mechanisms are required to provide information to management in a simple, consistent format and to forward information on important events for management purposes.

Monitoring and Tuning

The number of devices and the geographic disparity of devices in a netcentric environment increase the effort required to monitor the system. The number of events generated in the system rises due to the increased complexity. Devices such as client machines, network components, and servers generate events on startup or failure to periodically report device status.

Security

The security concerns of netcentric environments have been widely publicized. Although requirements for netcentric security architectures are constantly evolving as new security breaches are discovered, there are many tools categories that can help provide reasonable levels of security.

User Administration

The netcentric environment introduces many new challenges to the task of user administration. The majority of these stem once again from the dramatically increased number of system components. Adding a user to the system may require adding a user to the network, one or more server operating systems, one or more database systems (so that the user can access data), an e-mail system, and an existing host-based system.

Production Control

Scheduling processes across a distributed environment can be quite complex, requiring significant management effort to ensure that the processes run smoothly. Many other day-to-day activities become more difficult in a distributed environment, including print management, file transfer and control, mass storage management, backup and restore, archiving, and system startup and shutdown.

Help Desk

As netcentric computing puts the operations Help Desk closer to the "end user" in terms of visibility and influence, the Help Desk will need to become integrated with the business processes being supported through netcentric. Unless the operations Help Desk is well integrated with the business process, there is risk that the user may be given information that is incorrect, forwarded to the wrong department, or otherwise mishandled. It is also important that the information collected by the Help Desk about a user be properly shared with other stakeholders in the business process.

EXECUTION ARCHITECTURE

The netcentric Execution Architecture Framework identifies those common, run-time services required when an application executes in a netcentric environment. The services can be broken down into logical areas: Presentation Services, Information Services, Communication Services, Communication Fabric Services, Transaction Services, Environment Services, Base Services, and Business Logic (Exhibit 7).

As shown in the figure, the netcentric execution architecture is best represented as an extension to a client/server execution architecture. The figure shows the logical representation of a requester and a provider, designated by the "Client" and the "Server." Although the figure shows only one "Client" and one "Server," a physical implementation of an execution architecture typically has many clients and many servers. Thus, the services described here can be located on one physical machine, but most likely will span many physical machines, as shown in Exhibit 8.

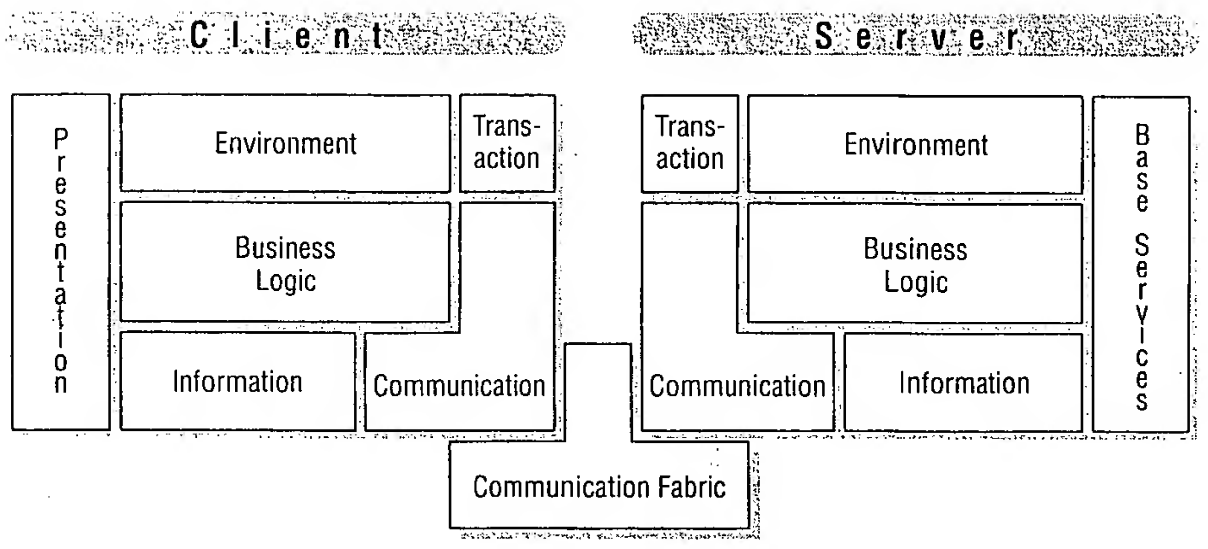


Exhibit 7. Netcentric Execution Architecture.

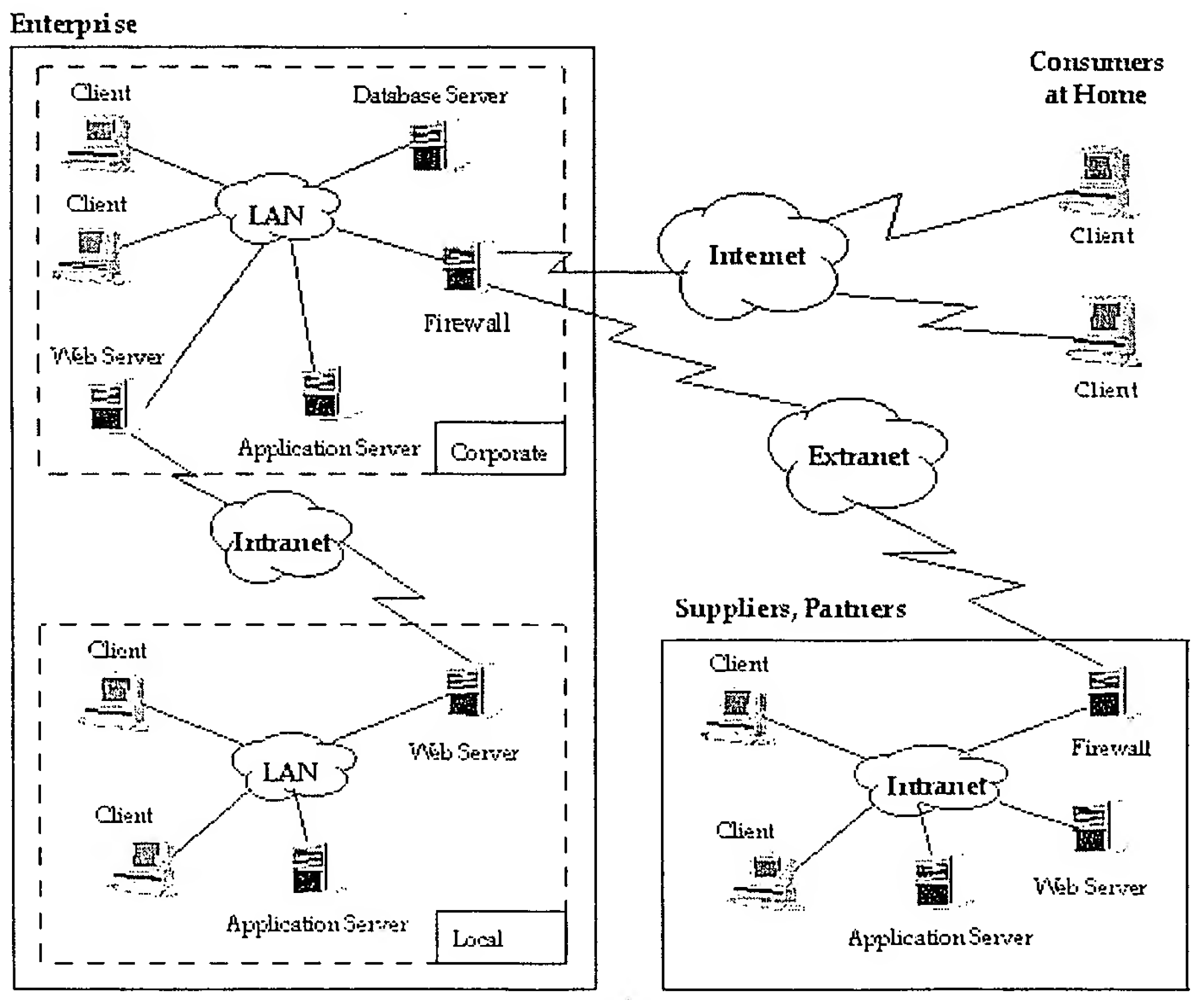


Exhibit 8. Execution Architecture: Physical Picture.

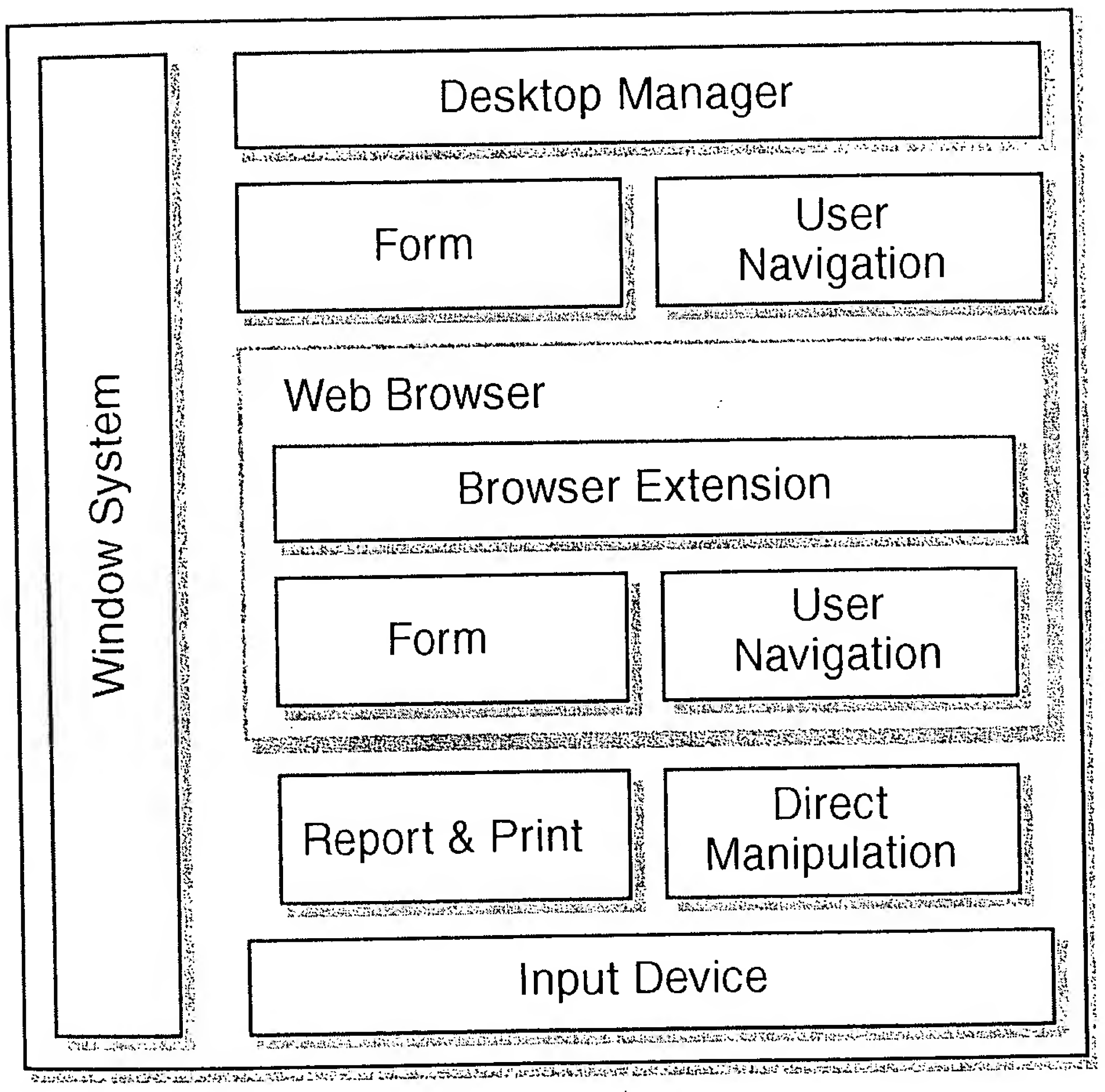


Exhibit 9. Presentation Services.

This section provides an overview of the services and subservices within the execution architecture. More detailed information is provided in the chapters devoted to each of these services in Section II.

PRESENTATION SERVICES

Presentation services (Exhibit 9) enable an application to manage the human–computer interface, including capturing user actions and generating resulting events, presenting data to the user, and assisting in the management of the dialog flow of processing. Typically, presentation services are required only by client workstations.

The major Presentation Services are

- Desktop Manager Services
- Direct Manipulation Services
- Form Services
- Input Devices Services
- Report and Print Services
- User Navigation Services
- Web Browser Services
- Window System Services

Desktop Manager Services

Desktop Manager Services provide for implementing the "desktop metaphor," a style of user interface that tries to emulate the idea of a physical desktop. It allows the user to place documents on the desktop, launch applications by clicking on a graphical icon, or discard files by dragging them onto a picture of a wastebasket. Desktop Manager Services include facilities for launching applications and desktop utilities and managing their integration.

Direct Manipulation Services

Direct Manipulation Services enable applications to provide a direct manipulation interface (often called "drag & drop"). A direct manipulation interface allows users to manage multiple "application objects" by manipulating visual representations of those objects. For example, a user may sell stock by dragging "stock" icons out of a "portfolio" icon and onto a "trading floor" icon. Direct Manipulation Services can be further divided into Display and Input/Validation.

Form Services

Form services enable applications to use fields to display and collect data. A field may be a traditional 3270-style field used to display or input textual data, or it may be a graphical field, such as a check box, a list box, or an image. Form services provide support for display, input/validation, mapping support, and field interaction management.

Input Devices

Input devices detect user input from a variety of input technologies, such as pen based, voice recognition, touch-screen, mouse, and digital camera.

Report and Print Services

Report and Print Services support the creation and on-screen previewing of paper or photographic documents, which contain screen data, application data, graphics, or images.

User Navigation Services

User Navigation Services provide a user with a way to access or navigate between functions within or across applications. Historically, this has been the role of a text-based menuing system that provides a list of applications or activities for the user to choose from. However, client/server technologies introduced new navigation metaphors. A common method for allowing a user to navigate within an application is to list available functions or information by means of a menu bar with associated pull-down menus or context-sensitive pop-up menus.

Web Browser Services

Web Browser Services allow users to view and interact with applications and documents made up of varying data types, such as text, graphics, and audio. These services also provide support for navigation within and across documents no matter where they are located through the use of links embedded into the document content. Web Browser Services retain the link connection, i.e., document physical location, and mask the complexities of that connection from the user.

Web Browser services can be further subdivided into

- Browser Extension Services
- Form Services
- User Navigation Services

Browser Extension Services

Browser Extension Services provide support for executing different types of applications from within a Browser. These applications provide functionality that extend Browser capabilities. The key Browser Extensions are plug-ins, helper/application viewers, Java applets, Active/X controls, and Java beans.

Form Services

Like Form Services outside the Web Browser, Form Services within the Web Browser enable applications to use fields to display and collect data. The only difference is the technology used to develop the Forms. The most common type of Forms within a browser is Hypertext Markup Language (HTML).

User Navigation Services

Like User Navigation Services outside the Web Browser, User Navigation Services within the Web Browser provide a user with a way to access or navigate between functions within or across applications. These User Navigation Services can be subdivided into three categories: Hyperlink, Customized Menu, and Virtual Reality.

Window System

Typically, part of the operating systems, Window System Services provide the base functionality for creating and managing a GUI: detecting user actions, manipulating windows on the display, and displaying information through windows and graphical controls.

INFORMATION SERVICES

Information Services (Exhibit 10) manage information assets and enable applications to access and manipulate data stored locally or remotely from documents, databases, or external data sources. They minimize an application's dependence on physical storage and location within the network. Information services may also be used directly by the end user when ad hoc data and document access are integral to the application work task. Information Services are grouped into two primary categories:

- Database Services
- Document Services

Database Services

Database services are responsible for providing access to a local or remote database as well as maintaining integrity of the data within the database. These services also support the ability to store data on either a single physical platform or in some cases across multiple platforms. These services are typically provided by database management system (DBMS) vendors and accessed via embedded or call-level SQL variants and supersets. Depending upon the underlying storage model, non-SQL access methods may be used instead.

Database Services include

- Storage Services
- Indexing Services
- Security Services
- Access Services
- Replication/Synchronization Services

Storage Services

Storage Services manage data physical storage. These services provide a mechanism for saving information so that data will live beyond program execution. Data are often stored in relational format (an RDBMS) but may also be stored in an object-oriented format (OODBMS) or other structures such as IMS and VSAM.

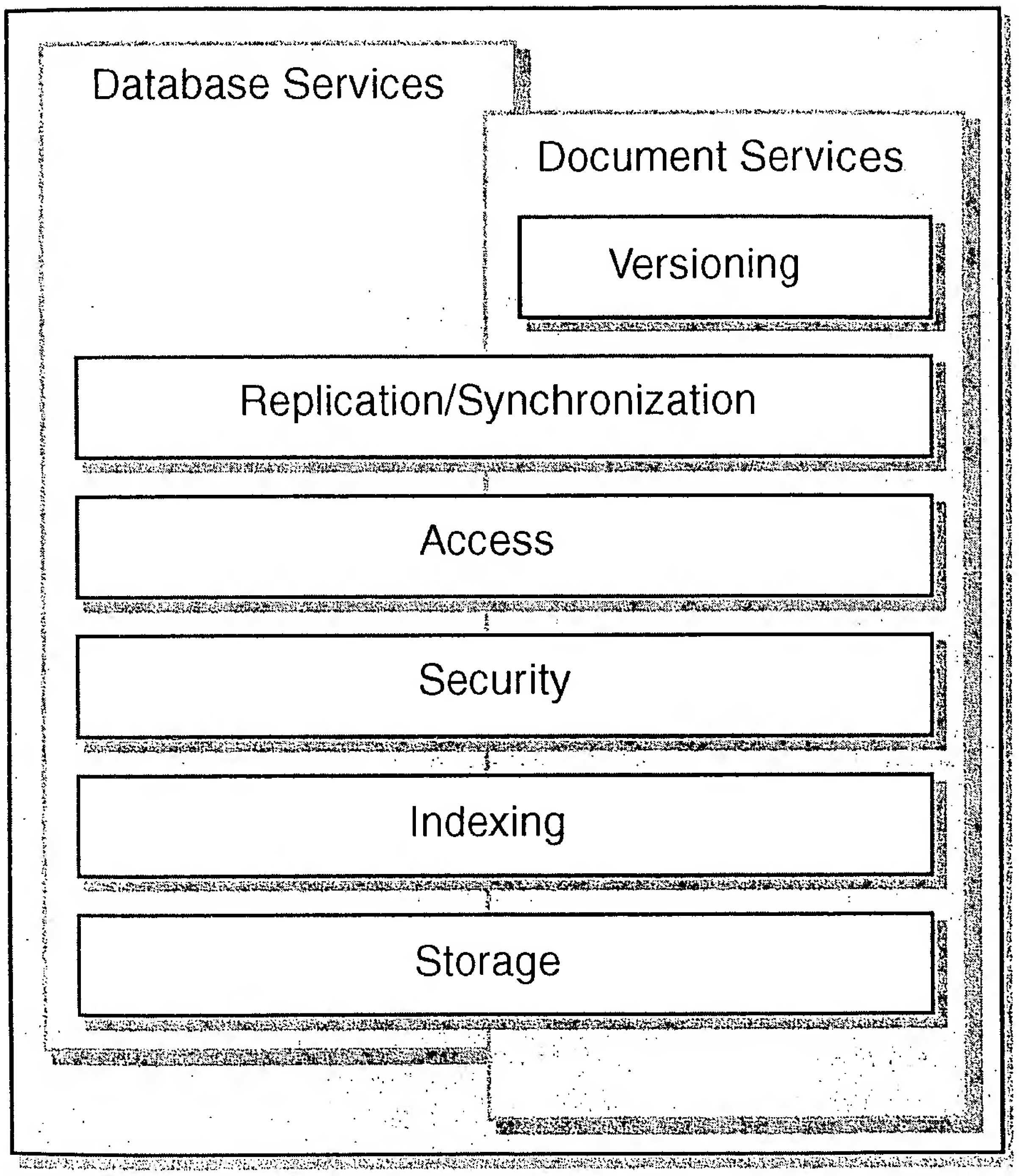


Exhibit 10. Information Services.

Indexing Services

Indexing Services provide a mechanism for speeding up data retrieval. In relational databases one or more fields can be used to construct the index. Therefore, when a user searches for a specific record rather than scanning the whole table sequentially the index is used to find the location of that record faster.

Security Services

Security Services enforce access control to ensure that records are only visible or editable by authorized people for approved purposes. Most DBMSs provide access control at the database, table, or row levels to specific users and groups as well as concurrency control. They also provide execution control for such things as stored procedures and database functions.

Access Services

Access Services enable an application to retrieve data from a database as well as manipulate (insert, update, or delete) data in a database. SQL is the primary approach for accessing records in today's DBMSs.

Replication/Synchronization Services

Replication Services support an environment in which multiple copies of databases must be maintained. Synchronization Services perform the transactions required to make one or more information sources that are intended to mirror each other consistent.

Document Services

Document Services provide similar structure and control for documents that DBMSs apply to record-oriented data. A document is defined as a collection of objects of potentially different types (e.g., structured data, unstructured text, images, or multimedia) that a business user deals with. Regardless of the software used to create and maintain the component parts, all parts together constitute the document, which is managed as a single entity.

Document Services include

- Storage Services
- Indexing Services
- Security Services
- Access Services
- Replication/Synchronization Services
- Versioning Services

Storage Services

Storage Services manage the physical storage of documents. Generally, the documents are stored in a repository using one of the following methods: proprietary database, industry standard database, or industry standard database and file system.

Indexing Services

Locating documents and content within documents is a complex problem and involves several alternative methods. Most document management products provide index services that support searching document repositories by the methods of attribute search, full-text search, context search, or Boolean search.

Security Services

Documents should be accessed exclusively through the document management backbone. If a document is checked in, checked out, routed, viewed, annotated, archived, or printed, it should be done only by authorized users. Security services controls access at the user, role, and group levels.

Access Services

Access Services support document creation, deletion, maintenance, and retrieval. These services allow users to capture knowledge or content through the creation of unstructured information, such as documents. Access Services also allow users to effectively retrieve documents that were created by them and documents that were created by others.

Replication/Synchronization Services

Replication Services support an environment in which multiple copies of documents must be maintained. Synchronization Services perform the transactions required to make one or more information sources that are intended to mirror each other consistent.

Versioning Services

These services maintain a historical record of the changes to a document over time. By maintaining this record, versioning services allow for the recreation of a document as it looked at any given point in time during its evolution.

COMMUNICATION SERVICES

Communication Services enable an application to interact transparently with other applications regardless of whether they reside on the same computer or on a remote computer.

There are five primary communications services categories (Exhibit 11):

- Core Messaging Services
- Specialized Messaging Services
- Communications Security Services
- Virtual Resource Services
- Directory Services

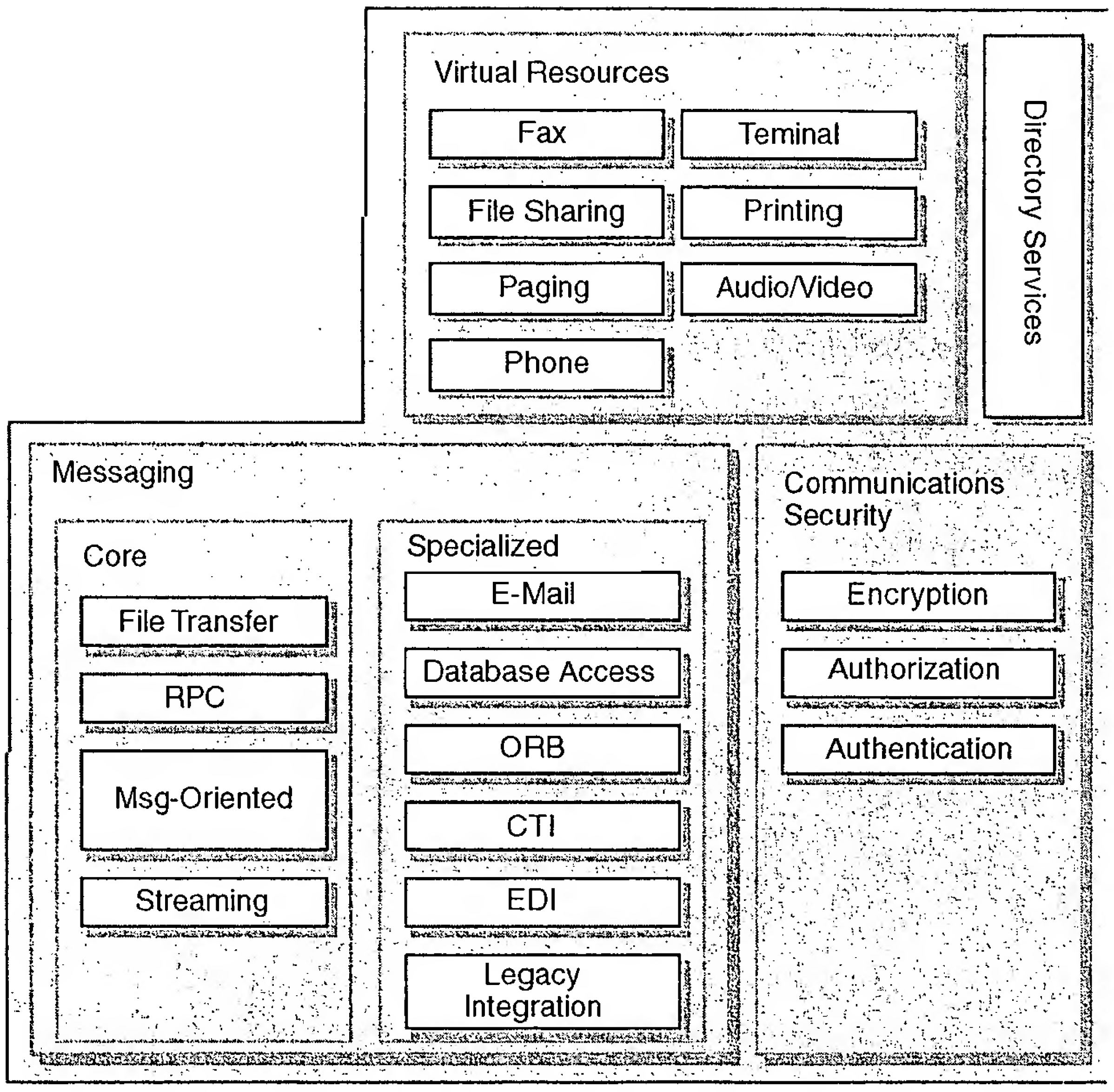


Exhibit 11. Communication Services.

Core Messaging Services

Broadly defined, messaging is sending information or commands between two or more recipients. Recipients may be computers, people, or processes within a computer. To send this message, a protocol (or in some cases, multiple protocols) is used that both the sender and receiver can understand. A protocol is a set of rules describing, in technical terms, how two end points should exchange information. Protocols exist at several levels during the exchange of information. Protocols facilitate transport of the message carrying the information. Both end points must recognize and observe the protocol. As an example, a common protocol in today's networks is the TCP/IP protocol.

Core messaging services can further be divided into the following services:

- *File Transfer Services.* File Transfer Services enable the copying and receiving of files or other large blocks of data between two resources.
- *Remote procedure call (RPC) services.* RPCs are a type of protocol by which an application sends a request to a remote system to execute a designated procedure using the supplied arguments and return the result.
- *Message-Oriented Services.* Message-Oriented Services refers to the process of distributing data and control through the exchange of records known as messages. Message-Oriented Services provide the application developer with a set of simple verbs (e.g., connect, send, receive, and disconnect) that are used to exchange information with other distributed applications.
- *Streaming Services.* Streaming is the process of transferring time-sensitive data streams (e.g., video and/or audio) in real time. Streaming differs from the other types of Core Messaging services in that it delivers a continuous, one-way stream of data, rather than the relatively short messages associated with RPC and Message-Oriented messaging or the large, batch transfers associated with File Transfer. Streaming may be used to deliver video, audio, and/or other real-time content across the Internet or within enterprise networks.

Specialized Messaging Services

Specialized Messaging Services extend the Core Messaging Services to provide additional functionality. Specialized Messaging Services may extend Core Messaging Services in the following general ways:

- Provides messaging among specialized systems by drawing upon basic messaging capabilities
- Defines specialized message layouts
- Defines specialized intersystem protocols
- Suggests ways in which messaging draws upon directory and security services to deliver a complete messaging environment

Specialized Messaging Services is comprised of the following subservices:

- *E-Mail Messaging.* E-Mail Messaging services reliably exchange messages using the store-and-forward messaging style. E-Mail message systems traditionally include a rudimentary form of directory services
- *Computer-Telephone Integration (CTI) Messaging.* CTI integrates computer systems and telephone systems to coordinate data and telephony activities. CTI Messaging has two primary functions: device-specific communication and message mapping.

- *EDI (Electronic Data Interchange) Messaging.* EDI supports system-to-system messaging among business partners by defining standard message layouts. Companies typically use EDI to streamline commercial transactions within their supply chains.
- *Object Request Broker (ORB) Messaging.* ORB Messaging enables objects to transparently make requests of and receive responses from other objects located locally or remotely. Objects communicate through an ORB. An ORB enables client objects to access server objects either locally or remotely over a network and invoke operations (i.e., functions and methods) on them.
- *Database Access Messaging.* Database Messaging services (also known as Database Access Middleware or DBAM) provide connectivity for clients to access databases throughout the enterprise.
- *Legacy Integration Messaging.* Legacy services provide gateways to mainframe legacy systems.

Communications Security Services

Communications Security Services control access to network-attached resources. Combining network Security Services with security services in other parts of the system architecture (e.g., application and database layers) results in robust security.

Communications Security Services are broken down into the following three categories:

- *Encryption Services.* Encryption services encrypt data prior to network transfer to prevent unauthorized interception.
- *Authorization Services.* When a user requests access to network resources, Authorization Services determines if the user has the appropriate permissions and either allows or disallows the access.
- *Authentication Services.* Authentication services verify network access requests by validating that users are who they claim to be. For secure systems, one or more authentication mechanisms can be used to validate authorized users and to verify which functions and data they have access to.

Virtual Resource Services

Virtual Resource Services proxy or mimic the capabilities of specialized, network-connected resources. This allows a generic network node to emulate a specialized physical device. In this way, network users can interface with a variety of specialized resources.

A common example of a Virtual Resource service is the capability to print to a network printer as if it were directly attached to a workstation.

Virtual Resource services include

- *Terminal Services.* Terminal services allow a client to connect to a non-local host via a network and to emulate the profile (e.g., the keyboard and screen characteristics) required by the host application.
- *Print Services.* Print services connect network workstations to shared printers.
- *File Sharing Services.* File Sharing Services allow users to view, manage, read, and write files that may be located on a variety of platforms in a variety of locations.
- *Phone Services.* Phone virtual resource services extend telephony capabilities to computer platforms.
- *Fax Services.* Fax Services provide for the management of both inbound and outbound fax transmissions.
- *Audio/Video Services.* Audio/Video Services allow nodes to interact with multimedia data streams. These services may be implemented as audio only, video only, or combined audio/video.
- *Paging Services.* Paging virtual resource services provide the message formatting and display functionality that allows network nodes to interface with wireless paging systems.

Directory Services

Managing information about network resources involves a variety of processes ranging from simple name/address resolution to the logical integration of heterogeneous systems to create a common view of services, security, etc. This breadth of functionality is discussed as part of Directory Services.

Because of their ability to unify and manage distributed environments, Directory Services play a key role in locating and accessing resources in a network, including Internet/intranet architectures.

COMMUNICATIONS FABRIC SERVICES

As communications networks become increasingly complicated and interconnected, the services provided by the network itself have by necessity increased as well. Clients and servers are rarely directly connected to one another but are commonly separated by a network of routers, servers, and firewalls, providing an ever-increasing number of network services such as address resolution, message routing, and security screening.

The Communications Fabric extends the client/server computing model by placing intelligence into the physical network, acknowledging the network as a sort of standalone system that provides intelligent shared network services. There is certainly overlap between services typically

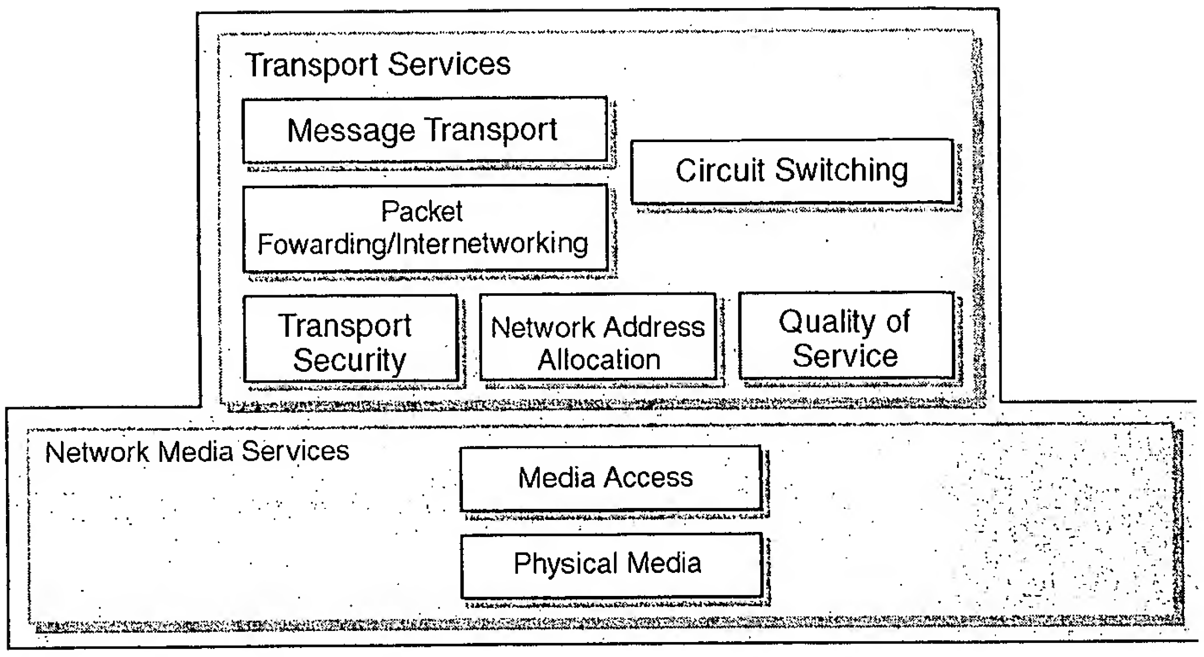


Exhibit 12. Communications Fabric Services.

thought of as part of a client/server architecture and services increasingly provided by the network itself.

Communications Fabric Services is comprised of two subservices: Transport Services and Network Media Services (Exhibit 12).

Transport Services

Transport Services are responsible for establishing, maintaining, and terminating end-to-end communications between users and processes. Connection management provides transfer services that ensure the delivery of data from sender to receiver, which support the transferring of messages from a process running on one machine to a process running on another machine. In addition, connection management provides services that initiate a connection, gracefully terminate a connection, and handle abrupt termination. These services take place for application before and after the data are formatted for transport over the network.

Transport Services include

- *Message Transport Services.* These are responsible for the end-to-end delivery of messages. They can include functionalities such as end-to-end data transfer, connection control, reliable transfer, flow control, and multiplexing.
- *Packet Forwarding/Internetworking Services.* The Packet Forwarding/Internetworking Service transfers data packets and manages the path that data take through the network. It includes functionalities

such as fragmentation/reassembly, addressing, routing, switching, and multicasting.

- *Circuit Switching Services.* Where Message Transport Services and Packet Forwarding/Internetworking Services support the transfer of packetized data, Circuit Switching Services establish physical circuits for the transfer of such things as circuit-switched voice, fax, and video.
- *Transport Security Services.* Transport Security Services (within the Transport Services layer) perform encryption and filtering.
- *Network Address Allocation Services.* Network Address Allocation Services manage the distribution of addresses to network nodes. This provides more flexibility compared to having all nodes assigned static addresses.
- *Quality of Service (QoS) Services.* QoS Services deliver a defined network throughput for designated traffic by allocating dedicated bandwidth, prioritizing data traffic, etc.

Network Media Services

The Network Media layer provides the following capabilities:

- Final framing of data for interfacing with the physical network
- Receiving, interpreting, and acting on signals from the communications fabric
- Transferring data through the physical network

Network Media Services performs two primary service functions:

- *Media Access Services.* Media Access Services manage the low-level transfer of data between network nodes. These services provide functions such as physical addressing, packet transfer, shared access, flow control, error recovery, and encryption.
- *Physical Media Services.* The Physical Media includes both the physical connectors and the the physical media (wired or wireless).

ENVIRONMENT SERVICES

Environment Services provide miscellaneous application and system level services that do not deal directly with managing the user interface, communicating to other programs, or accessing data (Exhibit 13).

Runtime Services

Runtime Services convert noncompiled computer languages into machine code during the execution of a program. Two subservices comprise Runtime Services: language interpreter and virtual machine.

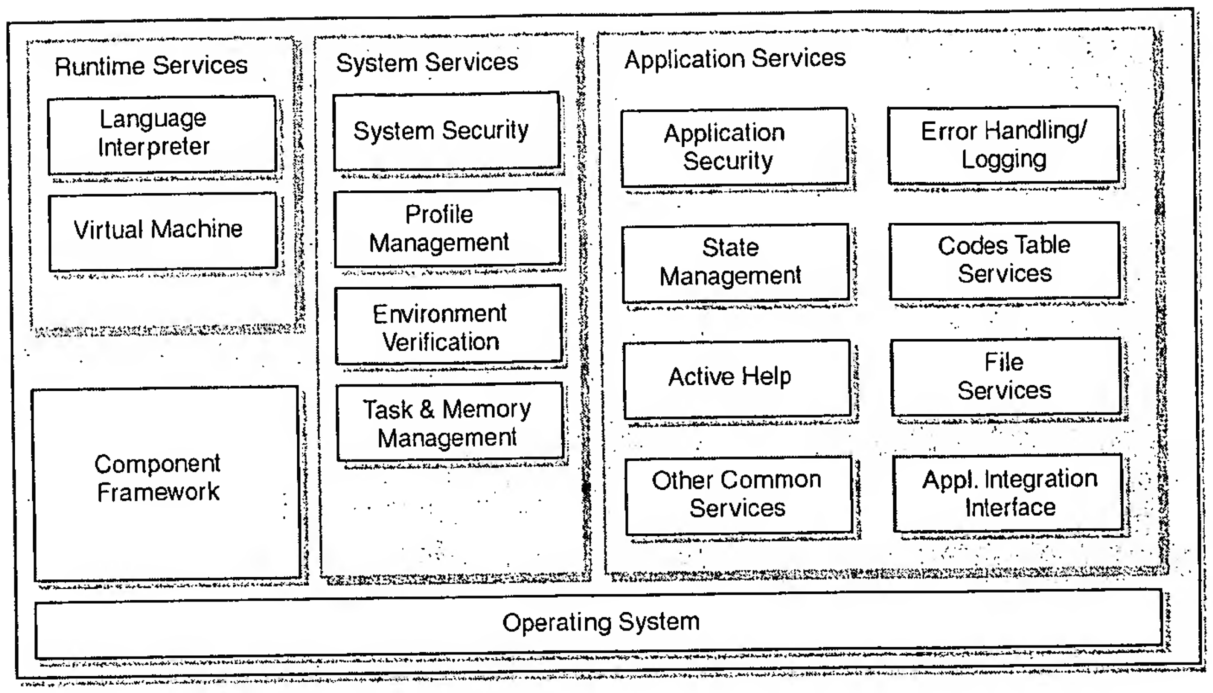


Exhibit 13. Environment Services.

- *Language Interpreter Services.* Language Interpreter Services decompose a fourth generation and/or a scripting languages into machine code (executable code) at runtime.
- *Virtual Machine Services.* Typically, a Virtual Machine is implemented in software on top of an operating system and is used to run applications. The Virtual Machine provides a layer of abstraction between the applications and the underlying operating system and is often used to support operating system independence.

System Services

System Services are services that applications can use to perform system-level functions. These services include

- System Security Services allow applications to interact with the operating system's native security mechanism. The basic services include the ability to login, logoff, authenticate to the operating system, and enforce access control to system resources and executables.
- Profile Management Services are used to access and update local or remote system, user, or application profiles. User profiles, for example, can be used to store a variety of information such as the user's language and color preferences to basic job function information that may be used by Integrated Performance Support or Workflow Services.
- Task and Memory Management Services allow applications and/or other events to control individual computer tasks or processes and

manage memory. They provide services for scheduling, starting, stopping, and restarting both client and server tasks (e.g., software agents).

- Environment Verification Services ensure functionality by monitoring, identifying, and validating environment integrity prior and during program execution. (e.g., free disk space, monitor resolution, and correct version).

Application Services

Application Services are miscellaneous services that applications can use for common functions. These common functions can apply to one application or can be used across applications. They include

- *Applications Security Services.* Besides system level security such as logging into the network, there are additional security services associated with specific applications, including user access services, data access services, and function access services.
- *Error Handling/Logging Services.* Error Handling Services support the handling of fatal and nonfatal hardware and software errors for an application. Logging Services support the logging of informational, error, and warning messages.
- *State Management Services.* State Management Services enable information to be passed or shared among windows and/or Web pages and/or across programs.
- *Codes Table Services.* Codes Table Services enable applications to utilize externally stored parameters and validation rules.
- *Active Help Services.* Active Help Services enable an application to provide assistance to a user for a specific task or set of tasks.
- *File Services.*
- *Application Integration Interface Services.* An Application Integration Interface provides a method or gateway for passing context and control of information to an external application.
- *Other Common Services.* This is a catch-all category for additional reusable routines useful across a set of applications (e.g., Date Routines, Time Zone Conversions, and Field Validation Routines).

Component Framework Services

Component Framework Services provide an infrastructure for building components so that they can communicate within an application and across applications, on the same machine or on multiple machines across a network, to work together. COM/DCOM and CORBA are the two leading component industry standards. These standards define how components should be built and how they should communicate.

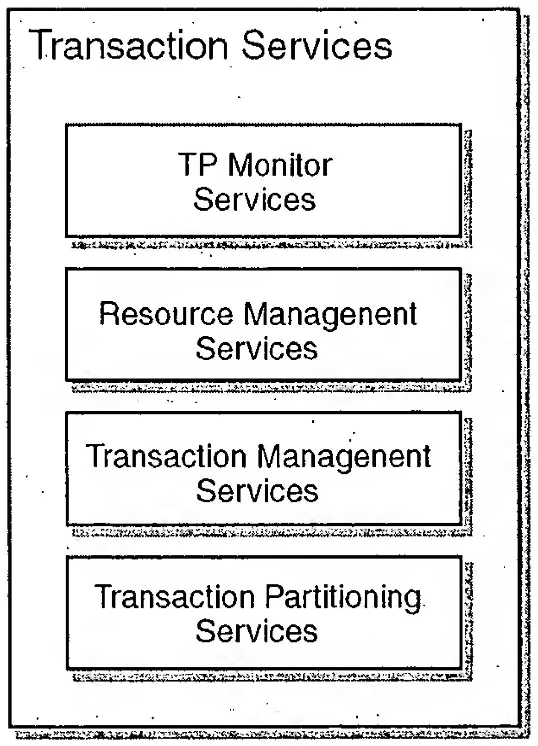

Transaction Services
TP Monitor Services
Resource Managenent Services
Transaction Managenent Services
Transaction Partitioning Services

Exhibit 14. Transaction Services.

Operating System Services

Operating System Services are the underlying services such as multitasking, paging, memory allocation, etc., typically provided by today's modern operating systems. Where necessary, an additional layer or APIs may be provided to gain either operating system independence or a higher level of abstraction for application programmers.

TRANSACTION SERVICES

Transaction Services provide the transaction integrity mechanism for the application. This allows all data activities within a single business event to be grouped as a single, logical unit of work.

In small- to moderate-scale environments of fewer than 150 simultaneous users on a single server, this service may be provided by the DBMS software with its restart/recovery and integrity capabilities. For larger client/server environments, an emerging class of software, referred to as "distributed on-line transaction managers," might be more applicable. These transaction managers provide sharing of server processes across a large community of users and can be more efficient than the DBMSs.

Transactions Services include (Exhibit 14):

- TP Monitor Services
- Resource Management Services

- Transaction Management Services
- Transaction Partitioning Services

TP Monitor Services

The Transaction Monitor Services are the primary interface through which applications invoke Transaction Services and receive status and error information. Transaction Monitor Services, in conjunction with Information Access and Communication Services, provide for load balancing across processors or machines and location transparency for distributed transaction processing.

Resource Management Services

A Resource Manager provides for concurrency control and integrity for a singular data resource (e.g., a database or a file system). Integrity is guaranteed by ensuring that an update is completed correctly and entirely or not at all. Resource Management Services use locking, commit, and rollback services and are integrated with Transaction Management Services.

Transaction Management Services

Transaction Management Services coordinate transactions across one or more resource managers either on a single machine or multiple machines within the network. Transaction Management Services ensure that all resources for a transaction are updated or, in the case of an update failure on any one resource, all updates are rolled back. This service allows multiple applications to share data with integrity.

Transaction Partitioning Services

Transaction Partitioning Services provide support for mapping a single logical transaction in an application into the required multiple physical transactions. For example, in a package or legacy-rich environment, the single logical transaction of changing a customer address may require the partitioning and coordination of several physical transactions to multiple application systems or databases. Transaction Partitioning Services provide the application with a simple single transaction view.

BASE SERVICES

Base Services provide support for delivering applications to a wide variety of users over the Internet, intranet, and extranet. Web Services include: Web Server Services, Push/Pull Services, Batch Services, Report Services, and Workflow Services (Exhibit 15).

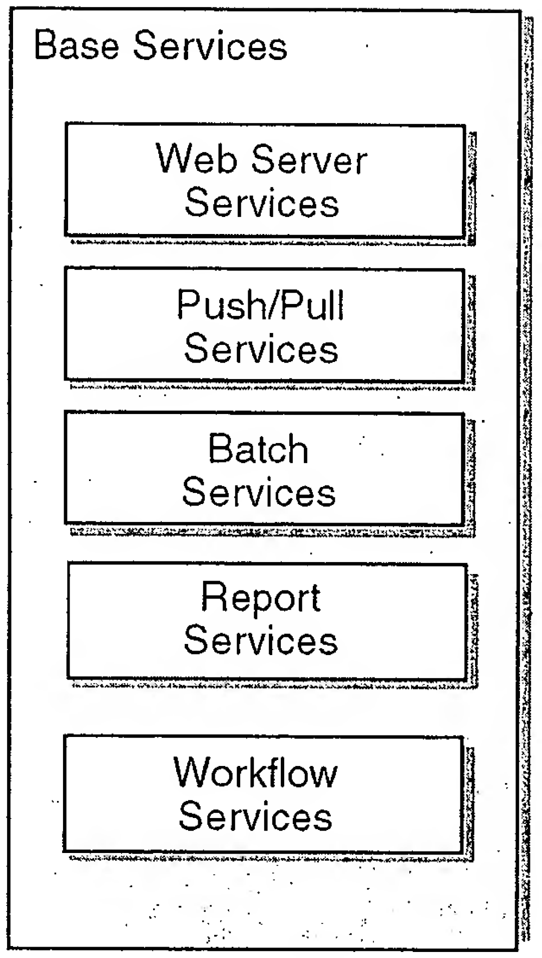


Exhibit 15. Base Services.

Web Server Services

Web Server Services enable organizations to manage and publish information and deploy netcentric applications over the Internet and intranet environments. These services support

- Managing documents in most formats such as HTML, Microsoft Word, etc.
- Handling of client requests for HTML pages
- Processing scripts such as Common Gateway Interface (CGI) or Active Server Pages (ASP)
- Caching Web pages

Push/Pull Services

Push/Pull Services allow for interest in a particular piece of information to be registered and then changes or new information to be communicated to the subscriber list. Depending upon requirements, synchronous or asynchronous push/pull services may be required. Synchronous push/pull services provide a mechanism for applications to be notified in real time if a subscribed item changes (e.g., a stock ticker). Asynchronous push/pull services do not require that a session-like connection be present between the subscriber and the information.

Batch Services

Batch processing is used to perform large-scale repetitive processing where no user involvement is required as well as reporting. Areas for design attention include scheduling, recovery/restart, use of job streams, and high availability (e.g., 24-hour running). In addition, close attention must be paid to performance as batch systems usually must be processed within strict batch windows.

Batch Services are comprised of the following subservices:

- *Driver Services.* These services provide the control structure and framework for batch programs. They are also referred to as Batch Scheduling Services.
- *Restart/Recovery Services.* These services are used to automatically recover and restart batch programs if they should fail during execution.
- *Batch Balancing Services.* These services support the tracking of run-to-run balances and totals for the batch system.
- *Report Services.* Project reporting tools are used to summarize and communicate information, using either printed paper or on-line report.

Report Services

Report Services are facilities for simplifying the construction and delivery of reports or generated correspondence. These services help to define reports and to electronically route reports to allow for on-line review, printing, and/or archiving. Report Services also support the merging of application data with predefined templates to create letters or other printed correspondence. Report Services include

- Driver Services
- Report Definition Services
- Report Built Services
- Report Distribution Services

Workflow Services

Workflow Services control and coordinate the tasks that must be completed to process a business event. Workflow enables tasks within a business process to be passed among the appropriate participants, in the correct sequence, and facilitates their completion within set times and budgets. Task definition includes the actions required as well as work folders containing forms, documents, images, and transactions. It uses business process rules, routing information, role definitions, and queues.

Workflow provides a mechanism to define, monitor, and control the sequence of work electronically. These services are typically provided by the server as they often coordinate activities among multiple users on multiple computers.

Workflow can be further divided into the following components:

- *Role Management Services.* These provide for the assignment of tasks to roles that can then be mapped to individuals.
- *Route Management Services.* These enable the routing of tasks to the next role.
- *Rule Management Services.* Rule Management Services support the routing of workflow activities by providing the intelligence necessary to determine which routes are appropriate given the state of a given process and knowledge of the organization's workflow processing rules.
- *Queue Management Services.* These services provide access to the workflow queues that are used to schedule work.

BUSINESS LOGIC

Business Logic is the core of any application, providing the expression of business rules and procedures (e.g., the steps and rules that govern how a sales order is fulfilled). As such, the Business Logic includes the control structure that specifies the flow for processing business events and user requests.

The execution architecture services described thus far are all generalized services designed to support the application's Business Logic. How Business Logic is to be organized is not within the scope of the execution architecture and must be determined based upon the characteristics of the application system to be developed. This section is intended to serve as a reminder of the importance of consciously designing a structure for Business Logic that helps to isolate the impacts of change and to point out that the underlying netcentric architecture is particularly well suited for enabling the packaging of Business Logic as components.

There are many ways in which to organize Business Logic, including rules-based, object-oriented, components, and structured programming. However, each of these techniques include common concepts, which we can group as Interface, Application Logic, and Data Abstraction (Exhibit 16).

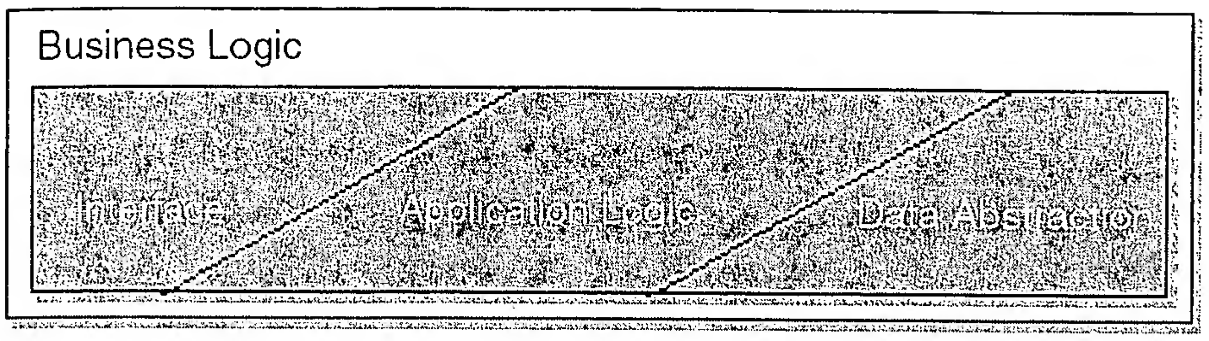


Exhibit 16. Business Logic.

Interface

Interface logic interprets and maps the actions of users into business logic processing activities. With the assistance of Presentation Services, Interface logic provides the linkage that allows users to control the flow of processing within the application.

Application Logic

Application Logic is the expression of business rules and procedures (e.g., the steps and rules that govern how a sales order is fulfilled). As such, the Application Logic includes the control structure that specifies the flow for processing for business events and user requests. The isolation of control logic facilitates change and adaptability of the application to changing business processing flows.

Data Abstraction

Information Access Services isolate the Business Logic from the technical specifics of how information is stored (e.g., location transparency, RDBMS syntax, etc.). Data Abstraction provides the application with a more logical view of information, further insulating the application from physical information storage considerations.

The developers of business logic should be shielded from the details and complexity of other architecture services (e.g., information services or component services), and other business logic for that matter.

It is important to decide whether the business logic will be separate from the presentation logic and the database access logic. Today, separation of business logic into its own tier is often done using an application server. In this type of an environment, although some business rules such as field validation might still be tightly coupled with the presentation logic, the majority of business logic is separate, usually residing on the server. It is also important to decide whether the business logic should be packaged as components to maximize software reuse and to streamline software distribution.

Another factor to consider is how the business logic is distributed between the client and the server(s) — where the business logic is stored and where the business logic is located when the application is being executed. There are several ways to distribute business logic:

1. Business logic can be stored on the server(s) and executed on the server(s).
2. Business logic can be stored on the server(s) and executed on the client.
3. Business logic can be stored and executed on the client.
4. Some business logic can be stored and executed on the server(s), and some business logic can be stored and executed on the client.

Having the business logic stored on the server enables developers to centrally maintain application code, thereby eliminating the need to distribute software to client machines when changes to the business logic occur. If all the business logic executes on the server, the application on the client will make requests to the server whenever it needs to execute a business function. This could increase network traffic, which may degrade application performance. On the other hand, having the business logic execute on the client may require longer load times when the application is initially launched. However, once the application is loaded, most processing is done on the client until synchronization with the server is needed. This type of an architecture might introduce complexities into the application that deal with the sharing of and reliance on central data across many users.

If the business logic is stored and executed on the client, software distribution options must be considered. Usually the most expensive option is to have a system administrator or the user physically install new applications and update existing applications on each client machine. Another option is to use a tool that performs automatic software distribution functions. However, this option usually requires the software distribution tool to be loaded first on each client machine. Another option is to package the application into ActiveX controls, utilizing the automatic install/update capabilities available with ActiveX controls — if the application is launched from a Web browser.

Currently, Internet applications house the majority of the business processing logic on the server, supporting the thin-client model. However, as technology evolves, this balance is beginning to shift, allowing business logic code bundled into components to be either downloaded at runtime or permanently stored on the client machine. Today, client-side business logic is supported through the use of Java applets, JavaBeans, Plug-ins and JavaScript from Sun/Netscape, and ActiveX controls and VBScript from Microsoft.

CONCLUSION

To operate optimally in the world of architectures, it is vital to remember a key point: one should not dwell too long at the abstract level. One can get mired in representations, in logical arguments. Pictures are important, but an architecture must be looked at pragmatically. It lives and breathes. It may evolve as the organization evolves. Yet, without the common understandings, common terminology, and common direction provided by architecture frameworks, project teams are putting their entire organization at risk.